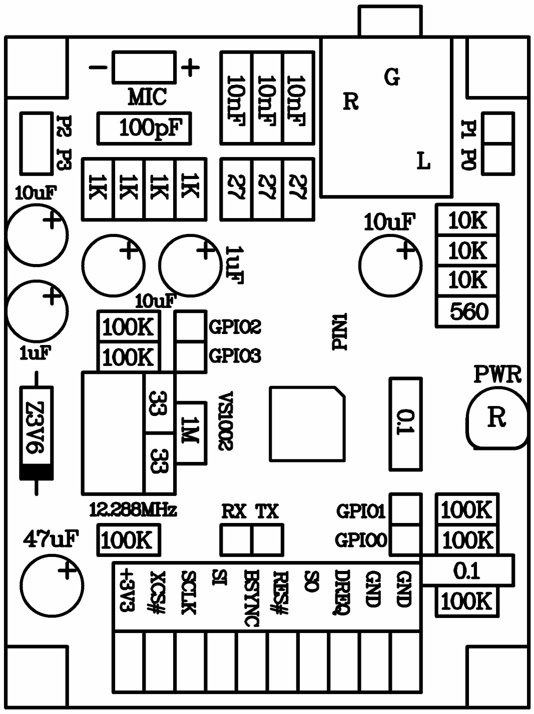
ET-MINI MP3



Board ET-MINI MP3 is a MP3 Decoder to convert signal into voice. ETT uses IC VLSI No.VS1002D to decode file MP3 and to be the permanent IC on board. VS1002D is the IC MP3 Decoder that is one of the highest quality and the easiest application. It can decode File MP3 and File WAVE instantly and there is Analog OUTPUT to be Stereo audio type, so we can connect it with headphone or amplifier instantly. Additionally, we can connect with Microcontroller easily by using SPI Serial Port standard. So, we can apply ET-MINI MP3 with Microcontrollers as desired.

The structural feature of Board ET-MINI MP3 is just a part of basic circuit VS1002D because it is not applied to be processed MP3 Player or connect with any version of Board Microcontroller but Board ET-MINI MP3 can apply with other project works independently. Concept of circuit is designed to connect VS1002D with necessary components such as circuit amplifier with Jack Stereo, it is quite convenient to connect with headphone or amplifier; and circuit Pre-Amplifier to receive signal from Microphone as Condenser type and including to circuit Crystal Oscillator Generator. Circuit VS1002D is arranged to be ready to active but there is no signal controller only. In the part of signal controller, there is Connector to connect with external microcontroller easily.

Specifications of Board ET-MINI MP3

- IC MP3 Decoder of VLSI No.VS1002D
- Be able to decode File MPEG that accesses decoder as MPEG 1.0 & 2.0 Audio layer III (CBR + VBR) and including of WAV and PCM
- Be able to access decode audio signal from Microphone to be standard ADPCM data
- Support Streaming Data for File MP3 or WAVE.
- Be able adjust Bass Control
- Active with signal Clock 12.288 MHz and can use X2 Mode internal PLL
- There is circuit to convert data into high quality of DAC voice with stereo Amplifier. We can connect Audio Out with amplifier or standard stereo headphone that has Impedance value about 30Ω instantly. Connector Audio Out of Board uses high quality Jack Stereo that can be interfaced with headphone or amplifier of computer PC instantly.
- Active with DC Voltage from 2.5V to 3.6V with LED to display status of Power and Zener Diode to protect over voltage
- Support connection between signal and Microcontroller through SPI Serial Port
- Be able to modify operation of board to be MP3 player as Standalone type without using the any controller from Microcontroller (see more information from "Application Note" of VLSI)
- Board size 4.4 x 5.6 cm.

Applications for Board ET-MINI MP3

We can apply Board ET-MINI MP3 for many types, especially the connection with Microcontroller. We can configure preferred conditions by self from program controller that is developed. We can apply it by using memories components such as SD/MMC or others to save File data for sending into VS1002D to decode and convert into voice. In this case, we do not mention about the memory management and File systems, so user must study and learn more information by self.

The well-known and easiest proceeding to connect Board ET-MINI MP3 with Microcontroller is connecting with SPI Serial Port. If Microcontroller is active with power supply from +2.5V to +3.3V, we can connect signal between Microcontroller and IC VS10002D of Board ET-MINI MP3 instantly; on the other hand, if Board Microcontroller is active with power supply +5V, we must find circuit to convert Logic +5V into Logic +3.3V first. ETT designs Board "ET-MINI LOGIC LEVEL SHIFTER" to support this application as shown in the diagram below.

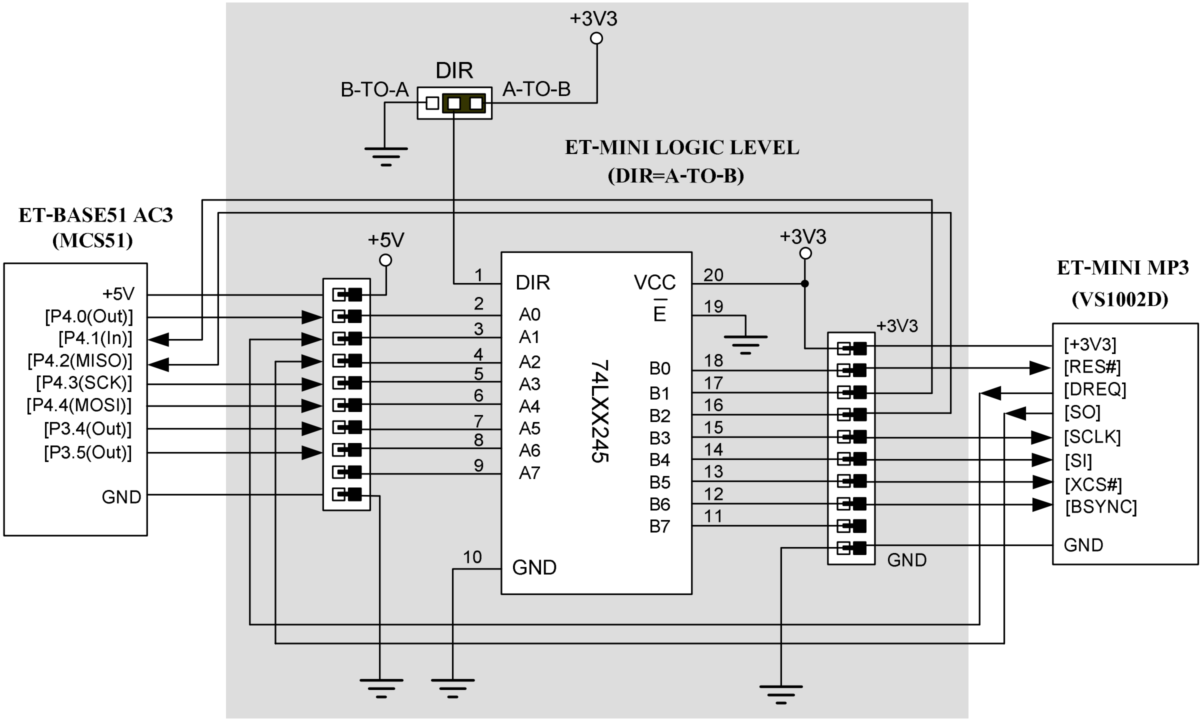
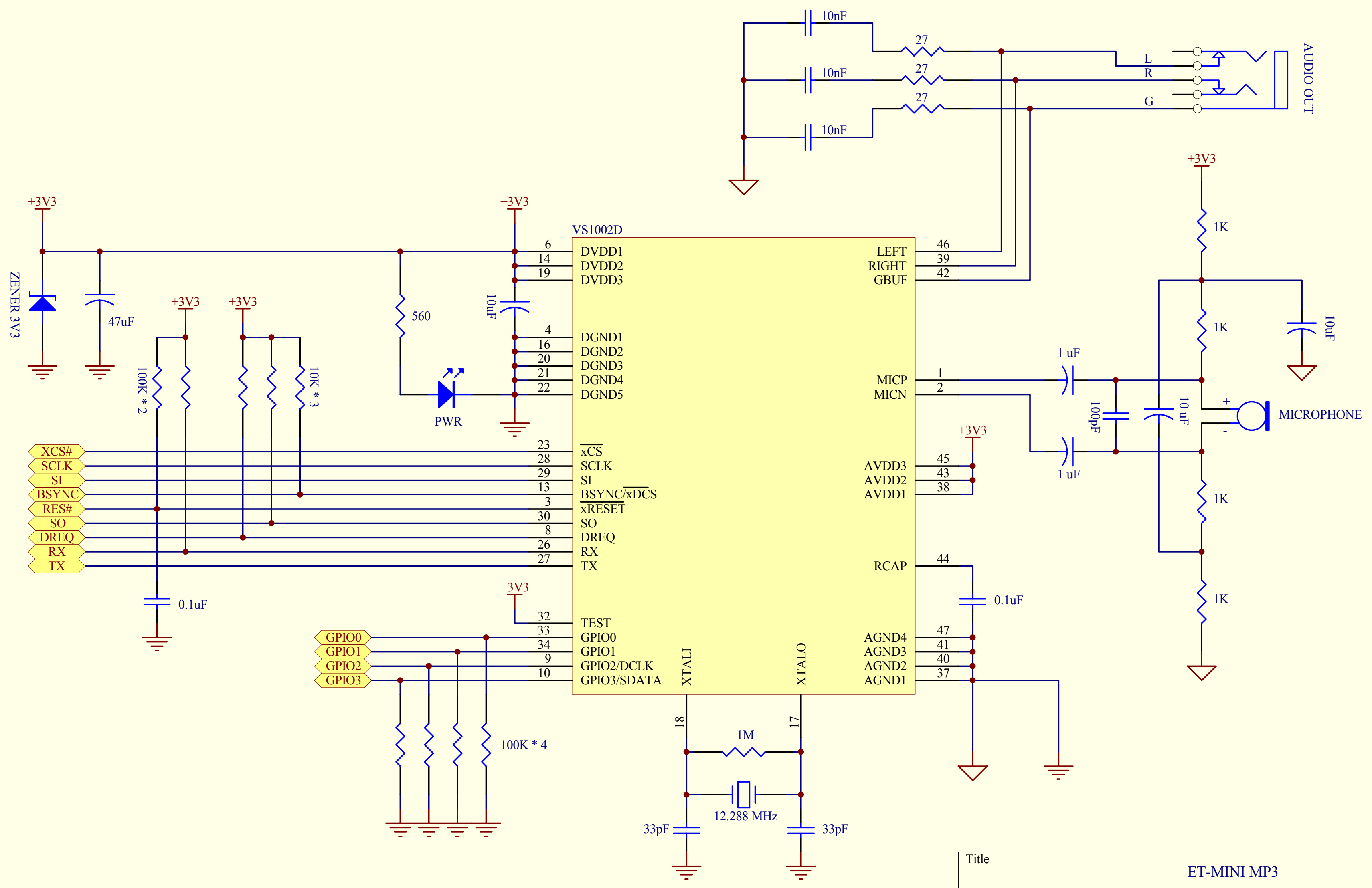


Figure displays the connection between board ET-MINI MP3 and Microcontroller that uses +5V Power Supply.



Title	ET-MINI MP3		
Size A4	Number WWW.ETT.CO.TH		Revision 1.0
Date:	21-Jun-2006	Sheet1 of 1	
File:	D:\My Circuit\MINI-MP3\MINI-MP3.Ddb	Drawn By:	Eakachai Makarn

An Example Connection ET-MINI MP3 with MCS51

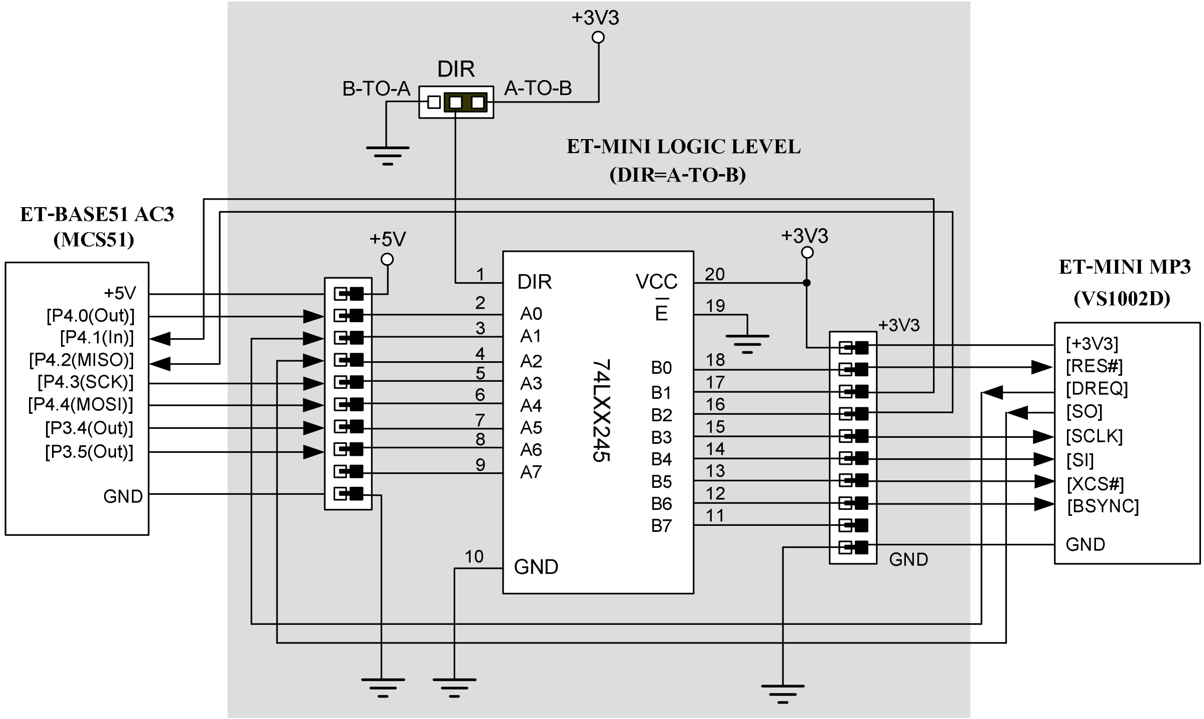
This example displays using Port SPI of MCS51 No.AT89C51AC3 to connect and command IC VS1002D that is a MP3 Decoder. The sample program will mention about proceeding to connect and command IC VS1002D to play music and voice only. We do not mention about the File systems management, so user must learn more how to read file data and send it into MP3 Decoder to convert into voice by self; for example, the proceeding to use Memory's types as SD/MMC memory or others.

In this case, we use Board Microcontroller MCS51 version ET-BASE51 AC3 to control operation of MP3 Decoder and use Board ET-MINI MP3. Remember, Board Microcontroller version ET-BASE51 AC3 actives with Voltage +5V; on the other hand, ET-MINI MP3 actives with Voltage +3V3, so IC VS1002D that is a MP3 Decoder in Board ET-MINI MP3 can not connect with signal Logic TTL with +5V signal level. So, we can not connect signal from both boards directly, we must find device to convert signal +5V into +3V3. In this case, we use ET-MINI LOGIC LEVEL SHIFTER for connection both boards. Additionally, it is necessary to use these devices as follows;

1. Board Microcontroller MCS51 version ET-BASE51 AC3
2. Board ET-MINI LOGIC LEVEL SHIFTER to convert signal Logic 5V into 3.3V
3. Board MP3 Decoder version ET-MINI MP3
4. +5V Adapter to supply power into Board ET-BASE51 AC3

In this case, we use 3 small File Wave that are not higher than 48 Kbytes totally and we use File Wave Format that we can hear it counting number from "0" to "2" in English version to store in Flash memory of AT89C51AC3 as Table type. Next, we must refer to memory position that stores the file and we must send it to MP3 Decoder to convert into voice as byte by byte respectively until it is completely. In this example, it converts File into data as Byte type and it is arranged as Table type in Flash memory. If it is C Language, it declares variable as Array type and saves it in Flash memory of CPU because it is more convenient to open file. This sample is written by C Language and uses Keil-C51 to interpret commands. Remember, C Language Program (Keil-C51) is used for Complier; if it just Demo Version, it can not interpret Source Code in this sample program because there is some restriction for using Program Keil-C51 Demo version that is not able to Complier Source Code higher than 2KByte. However, ETT provides Hex File that has already interpreted completely, so user can download "MCS51_MP3_PLAY_WAVE.HEX" into CPU instantly. This file is saved in Folder named "FINAL_HEX_TEST" to download into CPU of Board ET-BASE51 AC3 and can test it instantly. If everything is correct without any error after downloaded completely, when we press RESET and connect headphone or amplifier of computer PC with Board ET-MINI MP3 completely, we will hear counting number from 0 to 2 in English version and it will be repeated continuously.

The connection signal between boards



ET-BASE51 AC3 (MCS51)		Board ET-MINI LOGIC (DIR = A-TO-B)		Board ET-MINI MP3 (VS1002D)
[+5V]	→	[+5V] → [+3V3]	→	[+3V3]
[P4.0(Out)]	→	[A0] → [B0]	→	[RES#]
[P4.1(In)]	←	[B1] ← [A1]	←	[DREQ]
[P4.2(MISO)]	←	[B2] ← [A2]	←	[SO]
[P4.3(SCK)]	→	[A3] → [B3]	→	[SCLK]
[P4.4(MOSI)]	→	[A4] → [B4]	→	[SI]
[P3.4(Out)]	→	[A5] → [B5]	→	[XCS#]
[P3.5(Out)]	→	[A6] → [B6]	→	[BSYNC]
[GND]	↔	[GND] ↔ [GND]	↔	[GND]

show the directions and signal to connect MCS51 and MINI-MP3

An example Connection ET-MINI MP3 by AVR

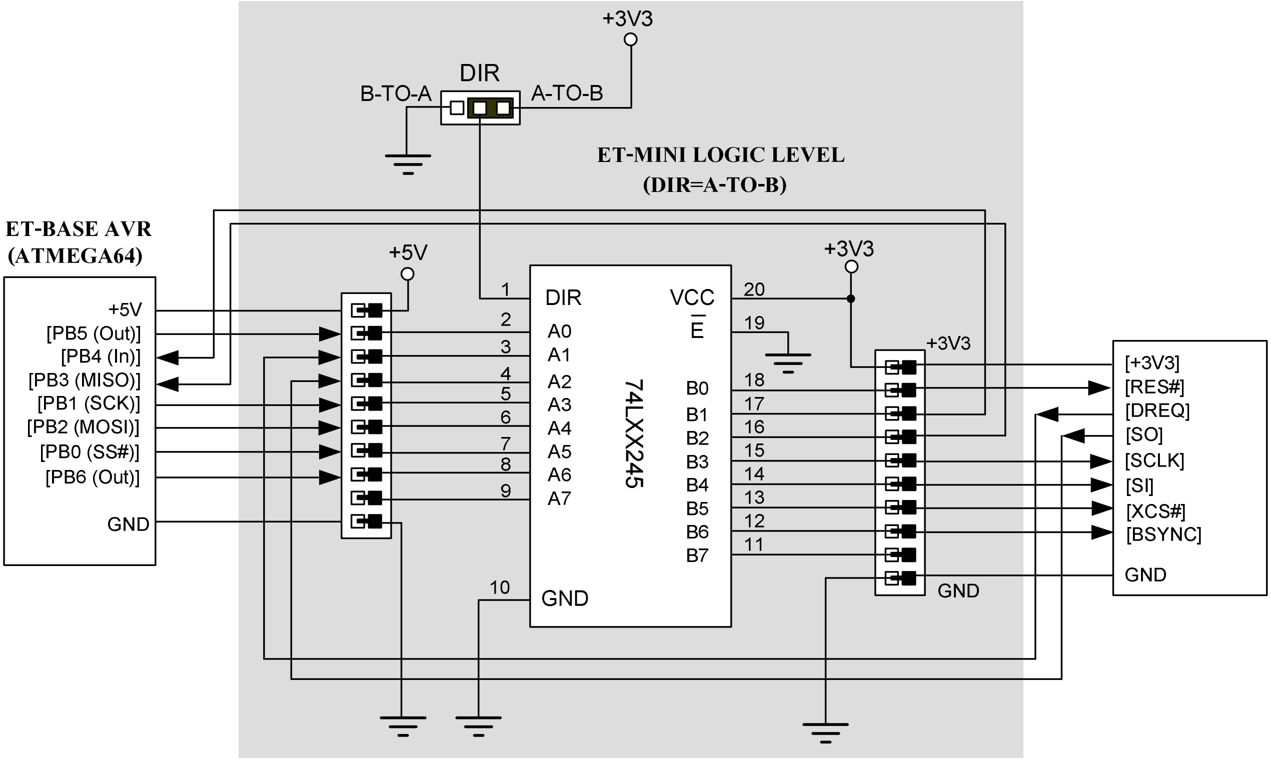
This example displays using Port SPI of AVR No.ATMEGA64 to connect and command IC VS1002D that is a MP3 Decoder. The sample program will mention about proceeding to connect and command IC VS1002D to play music and voice only. We do not mention about the File systems management, so user must learn more how to read file data and send it into MP3 decoder to convert into voice by self; for example, the proceeding to use Memory's types as SD/MMC memory or others.

In this case, we use Board Microcontroller AVR version ET-BASE AVR ATMEGA64 to control operation of MP3 Decoder and use Board ET-MINI MP3. Remember, Board Microcontroller version ET-BASE AVR ATMEGA64 runs with Voltage +5V; on the other hand, ET-MINI MP3 runs with Voltage +3V3, so IC VS1002D that is a MP3 Decoder in Board ET-MINI MP3 can not connect with signal Logic TTL with +5V signal level. So, we can not connect signal from both boards directly, we must find device to convert signal +5V into +3V3. In this case, we use ET-MINI LOGIC LEVEL SHIFTER for connection both boards. Additionally, it is necessary to use these devices as follows;

1. Board Microcontroller AVR version ET-BASE AVR ATMEGA64
2. Board to convert signal Logic 5V into 3.3V
3. Board MP3 Decoder version ET-MINI MP3
4. +5V Adapter to supply power into Board ET-BASE AVR

In this case, we use 3 small File Wave that are not higher than 48 Kbytes totally. For this example, we use File Wave Format that we can hear it counting number from "0" to "2" in English version to store in Flash memory of ATMEGA64 as Table type. Next, we must refer to memory position that stores the file and we must send it to MP3 Decoder to convert into voice as byte by byte respectively until it is completely. In this example, it converts File into data as Byte type and it is arranged as table type in Flash memory. If it is C Language, it declares variable as Array type and saves it in Flash memory of CPU because it is more convenient to open file. This sample is written by C Language and uses Code Vision AVR to interpret commands. Remember, C Language Program (Code Vision AVR) is used for Complier if it just Demo Version, it can not interpret Source Code in this sample program because there is some restriction for using Program Code Vision AVR version Demo that is not able to Complier Source Code higher than 2KByte. However, ETT provides Hex File that has already interpreted completely, so user can download into CPU instantly (ATMEGA64_MP3_PLAY_WAVE.HEX). This file is saved in Folder named "FINAL_HEX_TEST" to download into CPU of Board ET-BASE AVR ATMEGA64 and can test it instantly. If everything is correct without any error after downloaded completely, when we press RESET and connect headphone or amplifier of computer PC with Board ET-MINI MP3 completely, we will hear counting number in English version from 0 to 2 and it will be repeated continuously.

The connection signal between boards



ET-BASE AVR (ATMEGA64)		ET-MINI LOGIC LEVEL (DIR = A-TO-B)		ET-MINI MP3 (VS1002D)
[+5V]	→	[+5V] → [+3V3]	→	[+3V3]
[PB5(Out)]	→	[A0] → [B0]	→	[RES#]
[PB4(In)]	←	[B1] ← [A1]	←	[DREQ]
[PB3(MISO)]	←	[B2] ← [A2]	←	[SO]
[PB1(SCK)]	→	[A3] → [B3]	→	[SCLK]
[PB2(MOSI)]	→	[A4] → [B4]	→	[SI]
[PB0(SS#)]	→	[A5] → [B5]	→	[XCS#]
[PB6(Out)]	→	[A6] → [B6]	→	[BSYNC]
[GND]	↔	[GND] ↔ [GND]	↔	[GND]

show the directions and signal to connect MEGA64 and MINI-MP3

An Example Connection ET-MINI MP3 by ARM7

We will mention about an example using Port SPI of LPC2138/LPC2148 for connection IC VS1002D that is MP3 Decoder. The sample program will mention about proceeding to connect and command IC VS1002D to play music and voice only. We do not mention about the File systems management, so user must learn more how to read file data and send it into MP3 decoder to convert into voice by self; for example, the proceeding to use Memory's types as SD/MMC memory or others.

For this example, we use 11 small Wave File type and the memory of each file is not higher than 16 KB. We can hear it counting number from "0" to "10" and is saved in Flash memory of LPC2138/LPC2148. Next, we must refer to memory position that stores the file and we must send it to MP3 Decoder to convert into voice. In the first time, we intend to arrange File as Array table type and include them together in Code program because it is quite convenient to open file. Remember, C Language Program (Keil-C51) that is used for Complier is just Demo Version, so there is some restriction that does not Compiler Source Code is higher than 16KByte. Therefore, we should solve this problem by dividing Code program and Audio file first, so each file is not higher than 16KByte. After Files are interpreted into HEX completely, we must combine Code program and Audio File together. Structure of Memory space for saving Code program and code of Audio file are configured as follows;

Memory Position (Code)	Application
0x00000 - 0x03FFF (16KB)	Monitor Code Program
0x04000 - 0x07FFF (16KB)	Voice "Zero" (0f.wav)
0x08000 - 0x0BFFF (16KB)	Voice "One" (1f.wav)
0x0C000 - 0x0FFFF (16KB)	Voice "Two" (2f.wav)
0x10000 - 0x13FFF (16KB)	Voice "Three" (3f.wav)
0x14000 - 0x17FFF (16KB)	Voice "Four" (4f.wav)
0x18000 - 0x1BFFF (16KB)	Voice "Five" (5f.wav)
0x1C000 - 0x1FFFF (16KB)	Voice "Six" (6f.wav)
0x20000 - 0x23FFF (16KB)	Voice "Seven" (7f.wav)
0x24000 - 0x27FFF (16KB)	Voice "Eight" (8f.wav)
0x28000 - 0x2BFFF (16KB)	Voice "Nine" (9f.wav)
0x2C000 - 0x2FFFF (16KB)	Voice "Ten" (10f.wav)

Table shows the memory management for LPC2138/LPC2148

There are many proceedings to convert audio file into HEX; for example, using program Utility of EPROM programmer to open audio file as Binary File type and save in the Buffer of program. Next, we must save file that is converted into Hex file completely and we must configure Offset position value for storing data as 0x4000 up as shown in the table above. In this case, we will mention about using program Utility named "BIN2HEX" that is downloaded from Website of Keil and we can download this program free without any charge. Program BIN2HEX is a program that runs as Command Line and user can study User's Manual of program from HELP of program. While we running Program on Dos Prompt, program will display the proceeding. However, we create Batch File named "VOIC.BAT" to convert audio file into HEX File follow by the position address that is configured as in the table above, so it is quite convenient for user to apply. It is saved in Folder named "AudioData" that is overlapped internal Folder of C Language Source Code. Details of Batch File are described as follows;

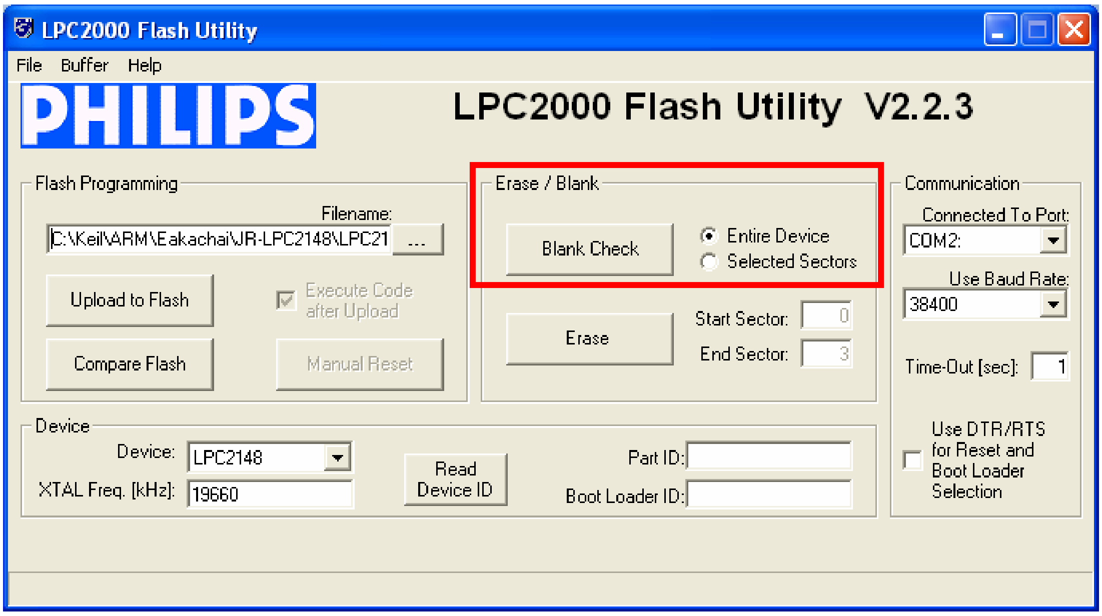
```
ECHO OFF
ECHO Generating VOICE.HEX with Wave Files...
DEL VOICE.HEX
BIN2HEX /L16384 /O16384  /4    /T /Q  0f.wav VOICE.HEX
BIN2HEX /L16384 /O32768  /4 /A /T /Q  1f.wav VOICE.HEX
BIN2HEX /L16384 /O49152  /4 /A /T /Q  2f.wav VOICE.HEX
BIN2HEX /L16384 /O65536  /4 /A /T /Q  3f.wav VOICE.HEX
BIN2HEX /L16384 /O81920  /4 /A /T /Q  4f.wav VOICE.HEX
BIN2HEX /L16384 /O98304  /4 /A /T /Q  5f.wav VOICE.HEX
BIN2HEX /L16384 /O114688 /4 /A /T /Q  6f.wav VOICE.HEX
BIN2HEX /L16384 /O131072 /4 /A /T /Q  7f.wav VOICE.HEX
BIN2HEX /L16384 /O147456 /4 /A /T /Q  8f.wav VOICE.HEX
BIN2HEX /L16384 /O163840 /4 /A /T /Q  9f.wav VOICE.HEX
BIN2HEX /L16384 /O180224 /4 /A    /Q  10f.wav VOICE.HEX
```

Figure displays commands in Batch File to interpret audio file into HEX.

When we open Batch File, we will get file named "VOICE.HEX" that is a part of 11 audio files and each file is arranged in memory as 16KB respectively as shown in the table above.

When we get both HEX files; Code Program (it is interpreted by Keil-CARM) and HEX File that is voice that is interpreted by Batch file (VOICE.BAT). Next step is downloading both Hex files into CPU by using program LPC2000 from Philips. In this case, there are two proceedings to do as follows;

1. Command to download Hex file as file by file. When we open Hex File that is audio file, must select type of delete memory as Enter Device that command to delete all memory and then command to download audio file first. Next, open HEX File of Source Code but we must select type of delete memory as Select Sectors type and must configure position Sector to be 0..3 and then command to download as in the picture below.



Filename: C:\Keil\ARM\Eakachai\JR-LPC2148\LPC21 ...

☑ Execute Code after Upload

● Entire Device ○ Selected Sectors

Start Sector: 0

End Sector: 3

Connected To Port: COM2:

Use Baud Rate: 38400

Time-Out [sec]: 1

□ Use DTR/RTS for Reset and Boot Loader Selection

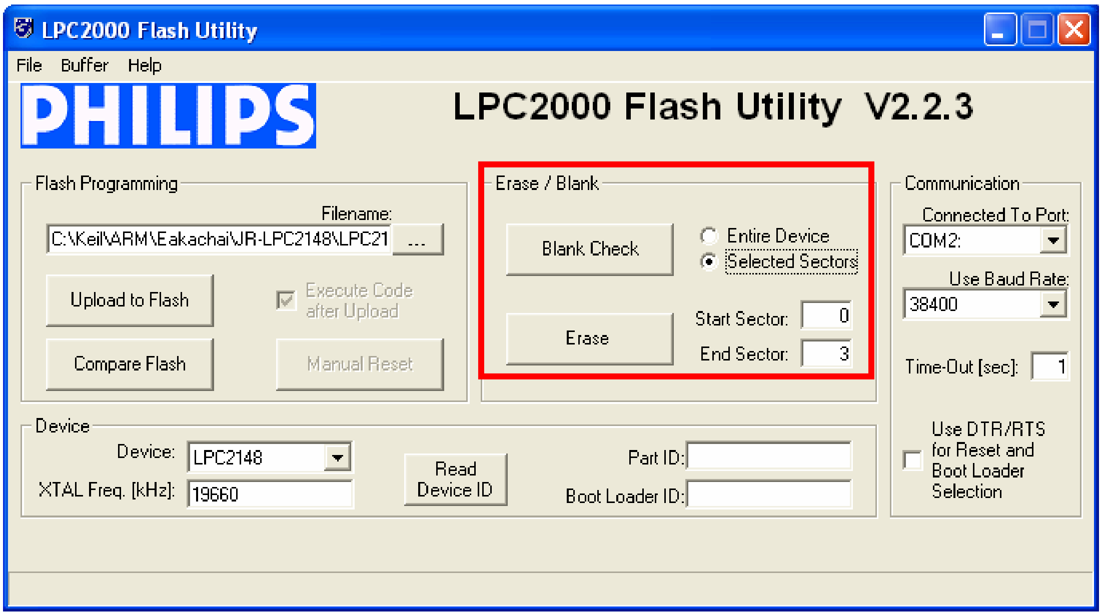
Device: LPC2148

XTAL Freq. [kHz]: 19660

Part ID:

Boot Loader ID:

↓



Filename: C:\Keil\ARM\Eakachai\JR-LPC2148\LPC21 ...

☑ Execute Code after Upload

○ Entire Device ● Selected Sectors

Start Sector: 0

End Sector: 3

Connected To Port: COM2:

Use Baud Rate: 38400

Time-Out [sec]: 1

□ Use DTR/RTS for Reset and Boot Loader Selection

Device: LPC2148

XTAL Freq. [kHz]: 19660

Part ID:

Boot Loader ID:

2. Must combine both 2 Hex Files into only one file and then command to download only one time as follows;

   a. Open Hex file of Source Code that is interpreted by Keil-CARM of Program Text Editor such as Notepad. Then command to delete the last line that is the end of file Intel HEX (End of HEX Record). The feature of the last line is shown as in the picture below.

```
:00000001FF
```

   b. Open HEX File of audio file that is interpreted by Batch File (VOIC.BAT). Next, copy all data in HEX file to place at the end of HEX File of Source Code and then save HEX File of Source Code that is combined together.
   c. Command to download HEX File that is combined together into board.

*****NOTE*****

For HEX File named that is combined together is "LPC2148_MP3_PLAY_WAVE.HEX" and "LPC2138_MP3_PLAY_WAVE.HEX" and is saved in Folder name "FINAL_HEX_TEST. This folder is overlapped internal Folder of C Language Source Code and Hex File that is in the same Folder of Source Code is Hex File of Source Code that does not combine with audio file. We can take Hex File in Folder named FINAL_HEX_TEST to download and test instantly. If everything is correct without any error after downloaded completely, when we press RESET and connect with headphone or amplifier of computer PC, we will hear the counting number from 0 to 10 in English version and it will be repeated continuously.

The connection signal between boards

Board CP-JR ARM7 LPC2138 CP-JR ARM7 USB-LPC2148		Board ET-MINI MP3 (VS1002D)
[+3V3]	↔	[+3V3]
[GPIO0.2(Input)]	←	[DREQ]
[GPIO0.3(Output)]	→	[RES#]
[GPIO0.4(SPI0-SCLK)]	→	[SCLK]
[GPIO0.5(SPI0-MISO)]	←	[SO]
[GPIO0.6(SPI0-MOSI)]	→	[SI]
[GPIO0.7(Output)]	→	[XCS#]
[GPIO0.8(Output)]	→	[BSYNC]
[GND]	↔	[GND]

Shows directions and signal of connection between ARM7 and MINI-MP3